МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ДЕКЛАРАТИВНЕ ПРОГРАМУВАННЯ

КОМП'ЮТЕРНИЙ ПРАКТИКУМ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського як навчальний посібник для здобувачів ступеня магістра за освітньою програмою «Комп'ютерний моніторинг та геометричне моделювання процесів і систем»

Київ
КПІ ім. Ігоря Сікорського
2019

Декларативне програмування: Комп’ютерний практикум : навч. посіб. для здобувачів ступеня магістра за освітньою програмою «Комп’ютерний моніторинг та геометричне моделювання процесів і систем» / КПІ ім. Ігоря Сікорського; уклад.: С.І. Шаповалова. – Електронні текстові дані (1 файл: 1,47 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2019. – 62 с.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 6 від 21.02.2019 р.) за поданням Вченої ради Теплоенергетичного факультету (протокол № 6 від 28.01.2019 р.)

Електронне мережеве навчальне видання

ДЕКЛАРАТИВНЕ ПРОГРАМУВАННЯ
КОМ’ЮТЕРНИЙ ПРАКТИКУМ

Укладач: *Шаповалова Світлана Ігорівна*, *канд. техн. наук, доцент*

Відповідальний редактор: *Сидоренко Ю.В., канд. техн. наук, доцент*

Рецензент: *Кондратюк В.А., канд. техн. наук, кафедра атомних електричних станцій та інженерної теплофізики*

За редакцією укладача

Посібник розроблений на підставі робочої програми кредитного модуля «Декларативне програмування» та призначений для якісної організації виконання комп’ютерного практикуму студентами.

Призначений для здобувачів ступеня магістра за освітньою програмою «Комп’ютерний моніторинг та геометричне моделювання процесів і систем».

Спрямований на формування у студентів умінь та досвіду програмування на мовах Prolog та Erlang. Забезпечує студентів необхідними прикладами виконання завдань, запланованих впродовж семестру.

ЗМІСТ

Вступ……. 4

Розподіл завдань по заняттям комп'ютерного практикуму…… 6

Завдання комп'ютерного практикуму……... 7

1 Представлення фраз Пролог-програми……. 7

2 Представлення родинних зв'язків……. 10

3 Арифметичні операції. Організація циклу……... 13

4 Розрахунок значень арифметичної функції в заданому інтервалі.. 16

5 Обробка списків…….. 18

6 Перестановка частин списків……. 21

7 Обробка матриці…….. 26

8 Використання предикатів збору…… 30

9 Обробка записів БД типу recorded…… 33

10 Задача класифікації……. 36

11 Розгалужені алгоритми……... 38

12 Конструювання списків…….. 41

13 Обробка елементів списків……. 43

14 Виокремлення частин списків…… 46

15 Обробка списку списків……. 50

16 Використання генераторів списків…… 54

17 Фільтрація списку……... 58

Рекомендована література……... 61

Інформаційні ресурси…….. 62

ВСТУП

Навчальна дисципліна «Декларативне програмування» входить до циклу професійної підготовки навчального плану магістрів спеціальності 122 «Комп`ютерні науки та інформаційні технології», освітньої програми «Комп'ютерний моніторинг та геометричне моделювання процесів і систем».

В дисципліні вивчаються концепції та загальні методи програмування на мовах Prolog і Erlang. Вибір мов програмування обумовлений тим, що:

- обидві є декларативними, але належать до двох різних типів – логічного (Prolog) і функціонального (Erlang) програмування;
- Prolog і Erlang мають однаковий базовий синтаксис.

Вивчення кредитного модуля спирається на знання, отримані за освітньою програмою підготовки бакалаврів за спеціальністю 122 «Комп'ютерні науки та інформаційні технології». Для сприйняття концепції логічного програмування необхідні базові знання з числення предикатів. Для програмування на мовах Prolog і Erlang також корисним є досвід у функціональному програмуванні.

Комп'ютерний практикум містить перелік завдань, в результаті виконання яких студенти отримають досвід застосування декларативних мов програмування, а саме:

- представлення та обробки структур даних, властивих мовам Prolog і Erlang;
- використання механізму пошуку з поверненням та засобів впливу на хід виконання Prolog-программи;
- вирішення задач з використанням функцій вищого порядку на мові Erlang;
- вирішення задач за допомогою недетермінованого програмування на мові Prolog.

Теоретичною основою щодо виконання комп'ютерного практикуму є курс лекцій з дисципліни „Декларативне програмування".

Для розв’язання практичних задач в навчальному посібнику наведено типовий приклад програми до кожного з завдань комп’ютерного практикуму.

Програмними засобами виконання завдань комп’ютерного практикуму є SWI Prolog та Erlang, реалізації яких є вільними з відповідними ліцензіями: Simplified BSD license та Erlang Public License. Інсталяції програмних засобів можна отримати з інформаційних ресурсів [1,2].

РОЗПОДІЛ ЗАВДАНЬ ПО ЗАНЯТТЯМ КОМП'ЮТЕРНОГО ПРАКТИКУМУ

N за-няття	N завд.	Завдання комп'ютерного практикуму	Програм-ний засіб	Кільк. год.
1	1	Представлення фраз Пролог-програми	Prolog	1
	2	Представлення родинних відношень		1
2	3	Арифметичні операції. Організація циклу		0,5
	4	Розрахунок значень арифметичних функцій у заданому інтервалі значень X		0,5
	5	Обробка списків		0,5
	6	Перестановка частин списків		0,5
3	7	Обробка матриці		1
		Контрольна робота		1
4	8	Використання предикатів збору		1
	9	Обробка записів БД типу recorded database		1
5	10	Задача класифікації	Erlang	1
	11	Розгалужені алгоритми		0,5
	12	Конструювання списків		0,5
6	13	Обробка елементів списків		1
	14	Виокремлення частин списків		1
7	15	Обробка списку списків		1
		Контрольна робота		1
8	16	Використання генераторів списків		1
	17	Фільтрація списку		1
9		*Залік*	Prolog Erlang	2

ЗАВДАННЯ КОМП'ЮТЕРНОГО ПРАКТИКУМУ

ЗАВДАННЯ № 1

Представлення фраз Пролог-програми

Метою є навчитись роботі в середовищі SWI Prolog.

Завдання: написати Пролог-програму, що складається з правила, вказаного в індивідуальному завданні, і бази даних, необхідній для отримання декількох відповідей на запити до цього правила.

Приклад виконання індивідуального завдання:

дві людини можуть спілкуватись, якщо вони володіють однією мовою.

Prolog -програма	**Приклади виконання запитів**
% ?Name1, ?Name2 *can_speak(X,Y):-* *know_language(X, Slang),* *know_language(Y, Slang),* *X\=Y.* *% ?Name, ?Language* *know_language(adam, engl).* *know_language(bob, engl).* *know_language(adam, rush).* *know_language(mike, engl).* *know_language(pete, rush).*	*?- can_speak(pete,X).* *X = adam ;* *No* *?- can_speak(adam,X).* *X = bob ;* *X = mike ;* *X = pete* *Yes* *?- can_speak(A,B).* *A = adam,* *B = bob ;* *A = adam,* *B = mike ;* *A = bob,* *B = adam ;* *A = bob,* *B = mike ;* *A = adam,* *B = pete ;*

	A = mike, *B = adam ;* *A = mike,* *B = bob ;* *A = pete,* *B = adam ;* *No*

Індивідуальні завдання

1. Два різні компоненти одягу можна одягнути одночасно, якщо вони відносяться до одного сезону і відповідають одному і тому ж типу погоди.
2. Двоє дітей одного віку можуть ходити в одну групу дитячого садку, якщо вони живуть в одному будинку.
3. Два спортсмени можуть брати участь в одному і тому ж змаганні, якщо вони займаються одним і тим же видом спорту і мають один і той же розряд.
4. У підборку картин для виставки можуть увійти три твори, що відносяться до одного художнього напряму і до однієї епохи створення.
5. У акваріум можна одночасно помістити риб трьох різновидів, якщо всі вони належать або не належать до хижаків.
6. Дві людини є братами, якщо вони - чоловіки і мають загальних батьків.
7. Дві кості доміно можуть бути встановлені поряд, якщо співпадає кількість крапок на дотичних половинках.
8. Два прикладні програмні продукти сумісні, якщо вони відкомпілювалися одним компілятором.
9. У збірку можуть увійти 3 літературних твори одного напряму, написані на одній мові.
10. Два лікарські препарати взаємозамінні , якщо вони мають одні й ті самі показання та протипоказання до застосування.
11. Два спортсмени можуть увійти до команди, якщо вони мають однаковий спортивний розряд і займаються різними видами спорту.

12. Дві футбольні команди різних країн можуть зустрітися в Єврокубку, якщо вони зайняли одне з перших чотирьох місць в національному чемпіонаті або виграли Кубок своєї країни.
13. Три квітки можуть скласти букет, якщо їх цвітіння відбувається в один і той же місяць.
14. Двоє тварин можуть одночасно знаходитися поряд на водопої, якщо обидві належать до травоїдних або м'ясоїдних.
15. У концерті можуть брати участь три музичні групи одного стилю і одного рівня виконання.
16. До збірки завдань контрольної роботи можуть увійти три завдання, які відносяться до різних розділів однієї і тієї ж навчальної дисципліни.
17. На ділянку можна посадити дерева трьох найменувань, якщо всі вони пристосовані до одного і тому ж клімату і є або листовими, або хвойними, або плодовими.
18. У стопку можна покласти 3 гральних карти різної гідності, але однакової масті.
19. У чвертьфіналі Ліги чемпіонів можуть зустрітися 2 футбольних команди з різних підгруп, якщо вони зайняли 1-е або 2-е місце в своїй підгрупі.
20. Двоє тварин можуть скласти пари, якщо вони протилежної статі і однієї породи. (Створити БД по котах або собаках.)
21. Два боксери можуть зустрітися в поєдинку, якщо вони в одній ваговій категорії і у них однаковий спортивний розряд.
22. Дві людини відносяться до одного знаку зодіаку, якщо вони народилися в один і той же місяць.
23. Фруктовий коктейль повинен складатися з трьох фруктів різних кольорів, які дозрівають в один і той же місяць.
24. Дві людини є близнятами, якщо у них одні і ті ж батьки і вони народилися в один і той же день.
25. У стопку можна покласти 3 гральних карти різної масті, але однакової гідності.

ЗАВДАННЯ № 2

Представлення родинних зв’язків

Метою є навчитись представляти природньомовні фрази на мові Prolog.

Завдання: описати на Пролозі відношення, що визначає споріднений зв'язок за індивідуальним завданням. У базі даних представити факти, що представляють одне або декілька з наступних відносин:

is_parent(?Ім’я батька/мами, ?Ім'я дитини)
is_female(?Ім'я),
is_male(?Ім'я),
marriage(?Ім'я чоловіка ? Ім'я жінки)

Припустимо використовувати лише зазначені структури відношень. Для представлення заданого відношення використовувати таку структуру аргументів:

is_relation(?Who ?Whom)

де <*relation*> відповідає заданому відношенню, Who визначає ім'я людини, що знаходиться у відношенні <*relation*> до людини на ім'я Whom. Наприклад, запит по відношенню «мачуха» (Мері є мачухою Боба):

is_stepmother(mary, bob).

Приклад виконання індивідуального завдання:

мачуха.

Prolog -програма	**Приклади виконання запитів**
% ?Who ?Whom *is_stepmother(X,Y):-* *marriage(F,X),* *is_parent(F,Y),* *not(is_parent(X,Y)).* *% ?Husbent, ?Wife*	*?- is_stepmother(X,Y).* *X = mary* *Y = bob* *Yes* *?- is_stepmother(Who,_).* *Who = mary ;*

marriage(mike,mary). *marriage(adam,ann).* *% ?Parent, ?Child* *is_parent(adam,dick).* *is_parent(ann,dick).* *is_parent(mike,bob).* *is_parent(mike,kate).* *is_parent(mary,kate).*	*No*

Для виконання завдання в БД необхідно у вигляді фактів представити відношення *marriage/2* та *is_parent*/2. Задане відношення представлено правилом *is_stepmother/2.*

Індивідуальні завдання

1. Свати (батьки чоловіка і дружини).
2. Двоюрідна бабуся.
3. Рідна сестра.
4. Свахи (матері чоловіка і дружини).
5. Двоюрідний брат.
6. Пасинок.
7. Невістка (дружина брата).
8. Кузини.
9. Троюрідна сестра.
10. Вітчим.
11. Свекор.
12. Дівер (брат чоловіка).
13. Теща.
14. Внучатий племінник.
15. Двоюрідний дядько.
16. Шурин (брат дружини).
17. Невістка (дружина сина).

18.Свекруха.

19.Зовиця (сестра дружини).

20.Зять.

21.Падчерка.

22.Двоюрідна тітка.

23.Троюрідний брат.

24.Єдинокровний брат (рідні по батькові).

25.Рідна сестра.

ЗАВДАННЯ № 3

Арифметичні операції. Організація циклу

Метою є навчитись реалізовувати на мові Prolog розгалужені алгоритми.

Завдання: представити на Пролозі функцію f(x), задану в індивідуальному завданні.

Приклад виконання індивідуального завдання:

$$f(x)=\begin{cases}0, x=0\\ \sum_{i=1}^{n} x*i, x>0,\end{cases}$$

Prolog –програма	**Приклад виконання запитів**
%+X,+N, -F(X) *func(0,N,N).* *func(X,N,F):-* *% sum_right(X,N,0,F).* *sum_left(X,N,F).* *%+X,+Counter,+Accumulator,-Sum* *sum_right(X,I,Acc,F):-* *I>=1,* *Acc1 is Acc+X*I,* *I1 is I-1,* *sum_right(X,I1,Acc1,F).* *sum_right(_,0,S,S).* *%+X,+Counter,-Sum* *sum_left(X,N,F):-* *N>=1,* *N1 is N-1,* *F1 is F+X*N,* *sum_left(X,N1,F1).* *sum_left(_,0,0).*	*?- func(2,3,F).* *F = 12 ;* *No*

В програмі наведено дві версії реалізації підсумовування відповідно з правою (*sum_right/4*) та лівою (*sum_left/3*) рекурсіями.

Індивідуальні завдання

1. $\begin{cases} x, & \text{якщо } 1<x<2, \\ \prod_{k=1}^{12} kx, & \text{якщо } x>2. \end{cases}$

2. $\begin{cases} 0, & \text{якщо } x<1 \\ x!, & \text{якщо } x>1. \end{cases}$

3. $\begin{cases} k, & \text{якщо } x<0, \\ \sum_{k=1}^{n} (kx+2), & \text{якщо } x>0. \end{cases}$

4. $\begin{cases} x!+x, & \text{якщо } x<55 \\ x, & \text{якщо } x>55. \end{cases}$

5. $\begin{cases} kx^k, & \text{якщо } x<10, \\ \sum_{k=1}^{8} kx, & \text{якщо } x>10. \end{cases}$

6. $\begin{cases} a, & \text{якщо } x<2 \\ (ax)!, & \text{якщо } x>2. \end{cases}$

7. $\begin{cases} (kx)!, & \text{якщо } 0<x<10, \\ x^k, & \text{якщо } x>1. \end{cases}$

8. $\begin{cases} 0, & \text{якщо } x<12 \\ \prod_{k=1}^{10} (kx+a), & \text{якщо } x>12. \end{cases}$

9. $\begin{cases} x^k, & \text{якщо } 7<x<9, \\ kx^k, & \text{якщо } x>9. \end{cases}$

10. $\begin{cases} 5, & \text{якщо } x<100 \\ \sum_{i=1}^{5} (ix-2), & \text{якщо } x>100. \end{cases}$

11. $\begin{cases} c+x, & \text{якщо } x<200, \\ x^n, & \text{якщо } x>200. \end{cases}$

12. $\begin{cases} 0, & \text{якщо } 1<x<2 \\ a+x!, & \text{якщо } x>2. \end{cases}$

13. $\begin{cases} 2, & \text{якщо } x<2, \\ x^n+x^m, & \text{якщо } x>2. \end{cases}$

14. $\begin{cases} 0, & \text{якщо } x<20 \\ a+x!, & \text{якщо } x>20. \end{cases}$

15. $\begin{cases} x^k + 15, & \text{якщо } 1<x<2, \\ \sum_{k=1}^{3} kx^k, & \text{якщо } x>2. \end{cases}$

16. $\begin{cases} x!, & \text{якщо } x>2 \\ \prod_{k=1}^{7} kx, & \text{якщо } 1<x<2. \end{cases}$

17. $\begin{cases} a, & \text{якщо } x>a, \\ \sum_{k=1}^{13} ax^k, & \text{якщо } x<a. \end{cases}$

18. $\begin{cases} k, & \text{якщо } 1<x<k \\ x!, & \text{якщо } x>k. \end{cases}$

19. $\begin{cases} x^n, & \text{якщо } x<n, \\ n, & \text{якщо } x>n. \end{cases}$

20. $\begin{cases} 0, & \text{якщо } x<n \\ \sum_{k=1}^{9} x^k, & \text{якщо } x>n. \end{cases}$

21. $\begin{cases} k, & \text{якщо } x<k, \\ \prod_{k=1}^{15} kx, & \text{якщо } x>k. \end{cases}$

22. $\begin{cases} n + x^n, & \text{якщо } 1<x<2 \\ 100, & \text{якщо } x>2. \end{cases}$

23. $\begin{cases} (x+n)^n, & \text{якщо } x<n, \\ nx^n, & \text{якщо } x>n. \end{cases}$

24. $\begin{cases} k, & \text{якщо } x<k \\ (x+k)!, & \text{якщо } x>k. \end{cases}$

25. $\begin{cases} c + x, & \text{якщо } x>10, \\ x^n, & \text{якщо } x<10. \end{cases}$

ЗАВДАННЯ № 4

Розрахунок значень арифметичної функції в заданому інтервалі

Метою є навчитись конструювати списки на мові Prolog.

Завдання: отримати список значень заданої в індивідуальному завданні функції. Інтервал значень змінної x, крок в інтервалі і додаткові параметри задаються в запиті.

Приклад виконання індивідуального завдання:

$$f(x) = \sum_{i=1}^{N} i * x$$

Prolog –програма	**Приклад виконання запитів**
%+X,+Final X,+Step Size,+N, *% -List of Function Values* *func_main(X,Xf,H,N,L):-* *Xf1 is Xf+H/100,* *func_list(X,Xf1,H,N,L).* *%+X,+Final X,+Step Size,+N,* *% -List of Function Values* *func_list(X,Xf,H,N,[F|Fs]):-* *X=<Xf,* *X1 is X+H,* *sum_right(X,N,0,F),* *func_list(X1,Xf,H,N,Fs).* *func_list(X,Xf,_,_,[]):-* *X>Xf.*	*?- func_main(1,2,0.2,3,X).* *X = [6, 7.2, 8.4, 9.6, 10.8, 12.0] ;* *No*

В зовнішній процедурі *func_main/5* вираховується межа інтервалу з урахуванням помилки накопичення. Процедура *func_list/5* безпосередньо будує список значень функції, які розраховуються *sum_right/4*, визначеною в лістінгі до завдання № 3.

Індивідуальні завдання

№ варіанту	Функція y(x)	№ варіанту	Функція y(x)
1	$\sum_{i=1}^{N}\frac{\cos(x+2i)}{i^2+1}-2x^2$	14.	$x+\frac{1}{x^N}+\prod_{i=1}^{N}\cos(*x)$
2	$(\sum_{i=1}^{N}\frac{1}{i^2})x\Big/\sum_{k=2}^{M}\frac{1}{k^2-1}$	15.	$\frac{tg^2x}{\sum_{i=1}^{N}i*x}$
3	$(\sum_{i=1}^{N}\frac{1}{x^2+i})+\sin x^2$	16.	$\frac{1+x}{1+x^3}*\prod_{i=3}^{N}\left(i^{\frac{1}{2}}-x\right)$
4	$\sum_{i=2}^{N}\frac{x^4-x^2}{i^2-1}\Big/x$	17.	$\sqrt{x+2}*\sum_{i=1}^{N}\frac{x^i}{i}$
5	$(\prod_{i=1}^{N}(i^2+x))x^2$	18.	$\frac{\prod_{i=1}^{N}(i+N)}{x+N}$
6	$\frac{1}{1+\cos x}\prod_{i=1}^{N}\frac{i^{-2}+0{,}5}{3}$	19.	$x^N*\sum_{i=1}^{N}\frac{i}{x+i}$
7.	$\left(\sum_{i=1}^{N}\frac{x^2}{i+x}\right)+tgx$	20	$\frac{x}{N}+\sum_{i=1}^{N}\frac{x+iM}{2i+1}$
8.	$\cos x^2*\prod_{i=1}^{N}(i^3+2.3*i)$	21	$\sqrt{x+1}\Big/\sum_{i=1}^{N}i\sin x$
9.	$\sqrt{x^2+5*x+2}*\sum_{i=1}^{N}\frac{1}{i*x^2}$	22	$\frac{N}{x}+\prod_{i=1}^{N}\frac{N+i+x}{i}$
10.	$\frac{x}{(+x)^2}\prod_{i=3}^{N}(+x)$	23	$\cos Nx\sum_{i=1}^{N}\frac{i}{N\sin x}$
11.	$(\sum_{i=1}^{N}\frac{1}{x+i})+tg(x^2)$	24	$N!\prod_{i=1}^{N}(x+i)$
12.	$\frac{\sin(x)}{x}*\prod_{i=2}^{N}\frac{(i-x)}{i}$	25	$\sqrt{x}+\sum_{i=1}^{N}x^i$
13.	$\sum_{i=1}^{N}\frac{x+i}{i!}-\cos x$		

ЗАВДАННЯ № 5

Обробка списків

Метою є навчитись обробляти елементи списків на мові Prolog.

Завдання: написати програму рішення вправи, вказаної в індивідуальному завданні.

Приклад виконання індивідуального завдання:

замінити n-й елемент списку на останній. Число n задається в запиті.

Prolog –програма	**Приклад виконання запитів**
%+List,+Counter,-New List, *%-Last Item* *replace([X|Xs],I,[X|Xs1],Xz):-* *I\=1,* *I1 is I-1,* *replace(Xs,I1,Xs1,Xz).* *replace([_|Xs],1,[Xz|Xs1],Xz):-* *replace(Xs,0,Xs1,Xz).* *replace([Xz],_,[Xz],Xz).*	*?- replace([1,2,3,4,5],3,L,Z).* *L = [1, 2, 5, 4, 5]* *Z = 5* *Yes*

В структурі процедури *replace/4* враховується аргумент, який буде зв’язаний останнім елементом заданого списку. Перша фраза процедури відтворює випадок перезапису в новий список всіх елементів, окрім n-го та останнього. Друга – встановлення останнього елементу замість n-го. Третя – закінчення рекурсії, якщо в списку залишився лише один елемент – останній.

Індивідуальні завдання

1. Видалити із списку елементи, що повторюються, залишивши один екземпляр.
2. Видалити передостанній і останній елементи списку і вставити їх відповідно першим і другим.

3. Додати в кінець заданого числового списку елемент, який є сумою 1-го і останнього елементів.
4. Видалити із списку всі елементи, що стоять на парних позиціях.
5. Заданий числовий список. Видалити з нього всі непарні числа.
6. Замінити n-й елемент списку на перший. Число n задається в запиті.
7. Поміняти місцями перший і останній елементи списку.
8. Знайти «серединний» елемент списку непарної довжини.
9. Заданий список непарної довжини. Поміняти місцями елемент, що стоїть до «серединного» елементу, з елементом, що стоїть після «серединного» елементу.
10. Видалити перші і останні три елементи заданого списку.
11. Заданий числовий список. Отримати список, в якому залишаться тільки цілі числа.
12. Вставити окремо заданий термін після «серединного» елементу списку непарної довжини.
13. Видалити із списку всі елементи, ідентичні заданому зразку, не залишаючи жодного екземпляра.
14. Видалити із списку всі елементи, що стоять на позиціях, кратних трьом.
15. Заданий числовий список. Отримати список, в якому залишаться лише парні числа.
16. Заданий числовий список непарної довжини. Знайти суму трьох його «серединних» елементів.
17. Додати в кінець заданого числового списку елемент, який є сумою 1-го і «серединного» елементів.
18. Додати в кінець заданого списку його другий і п'ятий елементи і видалити передостанній елемент.
19. Поміняти місцями n-й і (n+1)-й елементи списку. Число n задається в запиті.
20. Видалити із списку елементи, що все повторюються, не залишаючи жодного екземпляра.
21. Замінити кожен третій елемент заданого списку на останній.

22. Скласти список, що складається з елементів заданого списку, які більше обох своїх сусідів.
23. Отримати список, що складається з елементів заданого списку, що не повторюються.
24. Отримати список, що складаються з парних елементів заданого числового списку.
25. Заданий числовий список. Отримати список, в якому залишаться лише від’ємні числа.

ЗАВДАННЯ № 6

Перестановка частин списків

Метою є навчитись обробляти списки на мові Prolog.

Завдання: написати програму рішення вправи, вказаної в індивідуальному завданні.

Приклад виконання індивідуального завдання:

задано список, довжина якого кратна 4. Видалити третю чверть списку, а елементи другої розташувати в порядку, зворотному заданому.

Prolog -програма	**Приклад виконання запитів**
% ver 1 *%+List,-New List* *lists_pro_main1(L,L1):-* *length(L,N),* *K2 is N//4+1,* *K3 is N//2+1,* *K4 is 3*N//4+1,* *parts(L,1,K2,K3,K4,P1,P2,P4),* *reverse(P2,P2rev),* *append(P1,P2rev,P12),* *append(P12,P4,L1).* *parts([X|Xs],I,K2,K3,K4,[X|P1],P2,P4):-* *% 1st quarter* *I<K2,* *I1 is I+1,* *parts(Xs,I1,K2,K3,K4,P1,P2,P4).* *parts([X|Xs],I,K2,K3,K4,P1,[X|P2],P4):-* *% 2nd quarter* *I>=K2, I<K3,* *I1 is I+1,* *parts(Xs,I1,K2,K3,K4,P1,P2,P4).* *parts([_|Xs],I,K2,K3,K4,P1,P2,P4):-* *% 3rd quarter* *I>=K3, I<K4,*	*?- lists_pro_main1([1,2,3, a,b,c,10,20,30,100,200, 300],L).* *L = [1, 2, 3, c, b, a, 100, 200, 300]* *Yes*

```
    I1 is I+1,
    parts(Xs,I1,K2,K3,K4,P1,P2,P4).
parts(Xs,K4,_,_,K4,[],[],Xs).      % 4th quarter

                % ver 2
%+List,-New List
lists_pro_main2(L,L1):-
    length(L,N),
    K2 is N//4+1,
    K3 is N//2+1,
    K4 is 3*N//4+1,
    lists_pro(L,1,K2,K3,K4,[],L1).

%+List,+Counter,+K2,+K3,+K4,
%+Accumulator of 2nd quarter, -New List
lists_pro([X|Xs],I,K2,K3,K4,Acc,[X|Xs1]):-
                                    % 1st quarter
    I<K2,
    I1 is I+1,
    lists_pro(Xs,I1,K2,K3,K4,Acc,Xs1).
lists_pro([X|Xs],I,K2,K3,K4,Acc,Xs1):-
                                    % 2nd quarter
    I>=K2, I<K3,
    I1 is I+1,
    lists_pro(Xs,I1,K2,K3,K4,[X|Acc],Xs1).
lists_pro([_|Xs],I,K2,K3,K4,[X|Acc],[X|Xs1]):-
                                    % 3rd quarter
    I>=K3, I<K4,
    I1 is I+1,
    lists_pro(Xs,I1,K2,K3,K4,Acc,Xs1).
lists_pro(Xs,K4,_,_,K4,_,Xs).      % 4th quarter
```

Наведено 2 версії розв’язання задачі. В першій – список ділиться на відповідні частини *parts/8*, друга чверть обертається *reverse/2*, після цього відбувається злиття нового списку в потрібному порядку *append/3*.

В другій версії представлено оптимальне рішення задачі в одній процедурі *lists_pro/7* за 3/4n (де n - довжина списку) ітерацій. Під час проходження по списку:

- в першій чверті - елементи відразу переписуються в новий список,
- в другій чверті - накопичуються в акумуляторі в зворотному порядку,
- в третій чверті – з акумулятора переписуються в новий список,
- на першому елементі четвертої чверті рекурсія зупиняється і решта заданого списку повністю переноситься в новий.

Індивідуальні завдання

1. Задано список, довжина якого кратна 4. Видалити 1-у чверть списку, а елементи 3-ої розташувати в порядку, зворотному заданому.
2. Задано список, довжина якого кратна трьом. Отримати список, в якому перша і друга третини поміняються місцями.
3. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена остання третина.
4. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена перша третина, а елементи останньої третини розташовані в порядку, зворотному заданому.
5. Задано список, довжина якого кратна 4. Видалити останню чверть списку, а елементи першої розташувати в порядку, зворотному заданому.
6. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з третьої третини першого списку, другої третини другого і першої третини третього.
7. Задано 2 списки, довжина яких кратна 2. Отримати список, складений з половинок заданих списків в наступній послідовності: 1-а половина першого списку, 1-а половина другого списку, 2-а половина першого списку, 2-а половина другого списку.

8. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з першої третини першого списку, другої третини другого і третьої третини третього.
9. Задано список, довжина якого кратна трьом. Отримати список, в якому перша і остання третини збережуться, а елементи другої третини знаходитимуться в порядку, зворотному заданому.
10. Задано список, довжина якого кратна трьом. Отримати список, в якому елементи другої третини знаходяться в порядку, зворотному заданому.
11. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена остання третина.
12. Задано список, довжина якого кратна 4. Видалити 3-у чверть списку, а елементи 2-ої розташувати в порядку, зворотному заданому.
13. Задано список парної довжини. Отримати список, в якому перша і друга половини поміняються місцями.
14. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена друга третина.
15. Задано список парної довжини. Отримати список, в якому перша і друга половини поміняються місцями, причому елементи другої половини заданого списку йтимуть в порядку, зворотному заданому.
16. Задано 2 списки, довжина яких кратна 2. Отримати список, складений з половинок заданих списків в наступній послідовності: 2-а половина першого списку, 2-а половина другого списку, 1-а половина першого списку, 1-а половина другого списку.
17. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з першої третини першого списку, третьої третини другого і другої третини третього. Елементи останньої третини сформованого списку повинні розташовуватися в порядку, зворотному заданому.
18. Задано список, довжина якого кратна 4. Видалити 4-у чверть списку, а елементи 1-ої розташувати в порядку, зворотному заданому.

19. Задано список, довжина якого кратна трьом. Отримати список, в якому остання третина зберігається, а перша і друга міняються місцями. Причому елементи другої третини знаходитимуться в порядку, зворотному заданому.
20. Задано список, довжина якого кратна трьом. Отримати список, в якому елементи останньої третини знаходяться в порядку, зворотному заданому.
21. Задано 2 списки парної довжини. Отримати список, складений з другої половини першого заданого списку і першої половини другого.
22. Задано список, довжина якого кратна 4. Видалити 2-у чверть списку, а елементи 4-ої розташувати в порядку, зворотному заданому.
23. Задано список, довжина якого кратна трьом. Отримати список, в якому друга і остання третини поміняються місцями.
24. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з другої третини першого списку, першої третини другого і третьої третини третього.
25. Задано 2 списки, довжина яких кратна 2. Отримати список, складений з половинок заданих списків в наступній послідовності: 1-а половина першого списку, 1-а половина другого списку, 2-а половина першого списку, 2-а половина другого списку. Причому елементи других половин повинні бути розташовані в зворотному порядку по відношенню до заданого.

ЗАВДАННЯ № 7

Обробка матриці

Метою є навчитись обробляти списки списків на мові Prolog.

Завдання: виконати перетворення матриці або розрахунок її заданої характеристики. Якщо в завданні мова йде про діагоналі, задану матрицю вважати квадратною.

Приклад виконання індивідуального завдання:

Задана квадратна матриця невідомого розміру. Визначити суму елементів під побічною діагоналлю матриці (включно з елементами діагоналі).

Prolog –програма	**Приклади виконання запитів**
%+Test#, -Matrix *matrix_data(1,[[1,1,1,1,1],* *[1,1,1,1,1],* *[1,1,1,1,1],* *[1,1,1,1,1],* *[1,1,1,1,1]]).* *matrix_data(2,[[1,1,1,1,0],* *[1,1,1,0,0],* *[1,1,0,0,0],* *[1,0,0,0,0],* *[0,0,0,0,0]]).* *%+Test#, -Sum* *matrix_main(K,S):-* *matrix_data(K,Ms),* *length(Ms,N),* *matrix_pro(Ms,N,0,S).* *%+Matrix, +Rows Counter, +Accumulator, -Sum* *matrix_pro([R|Rs],I,Acc,S):-* *row_pro(R,1,I,S11),* *I1 is I-1,*	*?- matrix_main(1,X).* *X = 15* *Yes* *?- matrix_main(2,X).* *X = 0* *Yes*

Acc1 is Acc +S11, *matrix_pro(Rs,I1,Acc1,S).* *matrix_pro([],_,S,S).* *%+Row, +Counter, +N, -Row Sum* *row_pro([_|Xs],I,N,S):-* *I<N,* *I1 is I+1,* *row_pro(Xs,I1,N,S).* *row_pro([X|Xs],I,N,S):-* *I>=N,* *I1 is I+1,* *row_pro(Xs,I1,N,S1),* *S is S1+X.* *row_pro([],_,_,0).*	

В процедурі *matrix_pro/4* рекурсія відбувається по рядках матриці. На кожній ітерації обчислюється сума відповідних елементів поточного рядку – процедура *row_pro/4*. В *matrix_pro/4* реалізовано праву рекурсію, а в *row_pro/4* – ліву.

Індивідуальні завдання

1. Задана матриця невідомих розмірів. Отримати матрицю, в якій крайні елементи головної діагоналі поміняються місцями.
2. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій головна і побічна діагоналі поміняються місцями.
3. Задана квадратна матриця невідомого розміру. Отримати суму елементів, розташованих вище за головну діагональ матриці.
4. Задана квадратна матриця невідомого розміру. Отримати суму елементів третього квадранта матриці.
5. Задана матриця невідомих розмірів. Отримати матрицю, в якої половини головної діагоналі поміняються місцями.
6. Задана матриця розмірів MXN. Отримати матрицю, в якій поміняються місцями (M,1) -й і (1,1) -й елементи.

7. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на половинах діагоналей, розташованих праворуч від центру.
8. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на побічній діагоналі.
9. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій крайні елементи побічної діагоналі поміняються місцями.
10. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій поміняються місцями верхні половини головної і побічної діагоналей.
11. Задана матриця невідомих розмірів. Отримати суму центрального і всіх елементів, що оточують його.
12. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на головній діагоналі.
13. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями половини головної і побічної діагоналей, що знаходяться зліва від центру.
14. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій крайні елементи головної діагоналі поміняються місцями з сусідніми по діагоналі елементами.
15. Задана матриця невідомих розмірів. Отримати суму елементів першого квадранта матриці.
16. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій крайні елементи побічної діагоналі поміняються місцями з сусідніми по діагоналі елементами.
17. Задана матриця розмірів MXN. Отримати матрицю, в якій поміняються місцями (M,n) -й і (1,n) -й елементи.
18. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями центральний стовпчик і центральний рядок матриці.
19. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями половини головної і побічної діагоналей, що знаходяться праворуч від центру.

20. Задана матриця невідомих розмірів. Отримати суму елементів другого квадранта матриці.
21. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на половинах діагоналей, розташованих вище за центр.
22. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями нижні половини головної і побічної діагоналей.
23. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на повічній діагоналі.
24. Задана квадратна матриця невідомого розміру. Отримати суму елементів, розташованих вище за побічну діагональ матриці.
25. Задана матриця невідомих розмірів. Отримати суму елементів, що знаходяться на обох діагоналях.

ЗАВДАННЯ № 8

Використання предикатів збору

Метою є навчитись використовувати метапредикати.

Завдання: у базі даних міститися координати деякої множини точок. На підставі цих даних виконати індивідуальне завдання.

Представлення точки в БД повинне містити її позначення і координати.

Базу даних представити у вигляді фактів. Наприклад, для декартових координат може бути використане відношення наступної структури:

point(?Point_Sign, ?Abscissa, ?Ordinate).

Інформація має бути представлена в БД типу *assert/retract*. Записів повинно бути достатньо для отримання декількох варіантів рішення.

Приклад виконання індивідуального завдання:

по радіусу кола з центром в началі координат отримати всі точки БД, які не належать колу.

Prolog –програма	**Приклади виконання запитів**
:- dynamic *point/3.* *%?Point_Sign, ?Abscissa, ?Ordinate* *db_filling:-* *point(_,_,_),!.* *db_filling:-* *assert(point(a,1,1)),* *assert(point(b,2,2)),* *assert(point(c,1,3)).* *%+Radius, -Points List* *points_main(R,Xs):-* *db_filling,* *findall(X,inside(R,X),Xs).*	*?- points_main(2,L).* *L = [a]* *Yes* *?- points_main(5,L).* *L = [a, b, c]* *Yes*

%+Radius, -Point *inside(R,P):-* *point(P,X,Y),* *R*R >= X*X + Y*Y.*	

У випадку представлення даних у вигляді фраз Prolog-програми, необхідно такі фрази оголосити директивою *:- dynamic*. Процедура *db_filling*/0 виконує завантаження БД. В першій фразі здійснюється перевірка, чи була завантажена БД раніше. Якщо записи БД вже існують, завантаження припиняється. В іншому випадку відбувається завантаження початкової БД (фраза 2).

Індивідуальні завдання

1. По заданому центру і радіусу кола отримати всі точки БД, які лежать на колі.
2. По заданих великій і малій осях отримати всі можливі фокуси еліпсів.
3. Отримати всі можливі трикутники, які можуть бути побудовані на підставі точок БД.
4. По заданому центру і радіусу кола отримати всі точки БД, які лежать усередині кола.
5. На підставі точок БД отримати всі трикутники з рівною площею.
6. По заданих центрах і радіусах двох кіл знайти всі точки, які належать області перетину цих кіл.
7. По заданому радіусу отримати всі точки БД, які можуть бути центрами кіл, що не перетинаються.
8. Отримати всі можливі ромби, які можуть бути побудовані на підставі точок БД.
9. По заданій вершині і параметру p параболи з вертикальною віссю отримати всі точки БД, що належать параболі.
10. По заданих великій і малій осях отримати всі можливі центри еліпсів.

11. По заданих центру, дійсній і уявній осям отримати всі точки БД, що належать гіперболі.
12. Отримати всі можливі квадрати, які можуть бути побудовані на підставі точок БД.
13. По заданому радіусу отримати всі пари центрів кіл, всередину яких потрапляє однакова кількість точок БД.
14. Отримати всі можливі рівносторонні трикутники, які можуть бути побудовані на підставі точок БД.
15. Отримати всі можливі прямокутні трикутники, які можуть бути побудовані на підставі точок БД.
16. По заданій вершині і параметру p параболи з вертикальною віссю отримати всі точки БД, які знаходяться у внутрішній області параболи.
17. Отримати всі можливі паралелограми, які можуть бути побудовані на підставі точок БД.
18. По заданих центрах і радіусах двох кіл знайти всі точки, що належать диз'юнктивній сумі цих кіл.
19. По заданих центру і радіусам двох концентричних кіл знайти всі точки, що належать кільцю, утвореному заданими колами.
20. По заданих центру, великій і малій осям отримати всі точки БД, що знаходяться поза еліпсом.
21. Отримати всі можливі трапеції, які можуть бути побудовані на підставі точок БД.
22. На підставі точок БД отримати всі рівнобедрені трикутники.
23. По заданих центрах і радіусах двох кіл знайти всі точки, що належать об'єднанню цих кіл.
24. Отримати всі можливі трикутники, периметр яких буде менший заданого.
25. По заданій вершині і параметру p параболи з вертикальною віссю отримати всі точки БД, які знаходяться в зовнішній області параболи.

ЗАВДАННЯ № 9

Обробка записів БД типу recorded

Метою є навчитись веденню БД типу *recorded*.

Завдання: у базі даних міститися координати деякої множини точок. На підставі цих даних виконати індивідуальне завдання.

Представлення точки в БД повинне містити її позначення і координати.

Базу даних представити у вигляді фактів. Наприклад, для декартових координат може бути використане відношення наступної структури:

point(?Point_Sign, ?Abscissa, ?Ordinate).

Інформація має бути представлена в БД типу *recorded*. Записів повинно бути достатньо для отримання декількох варіантів рішення.

Приклад виконання індивідуального завдання:

по радіусу кола з центром в началі координат отримати всі точки БД, які не належать колу.

Prolog –програма	**Приклади виконання запитів**
%?Point_Sign, ?Abscissa, ?Ordinate *db_rec_filling:-* *recorded(db,point(_,_,_)),!.* *db_rec_filling:-* *recordz(db,point(a,1,1)),* *recordz(db,point(b,2,2)),* *recordz(db,point(c,1,3)).* *%+Radius, -Points List* *rec_points_main(R,Xs):-* *db_rec_filling,* *findall(X,inside1(R,X),Xs).* *%+Radius, -Point* *inside1(R,P):-* *recorded(db,point(P,X,Y)),*	*?- rec_points_main(1,L).* *L = []* *Yes* *?- rec_points_main(2,L).* *L = [a]* *Yes* *?- rec_points_main(3,L).* *L = [a, b]* *Yes* *?- rec_points_main(5,L).*

*R*R >= X*X + Y*Y.*	*L = [a, b, c]* *Yes*

Представлення даних типу *recorded* не потребує додаткових оголошень. Процедура *db_rec_filling*/0 виконує завантаження БД аналогічно процедурі *db_filling*/0, яка описана в завданні 8.

Індивідуальні завдання

1. По заданому центру і радіусу кола отримати всі точки БД, які лежать на колі.
2. По заданих великій і малій осях отримати всі можливі фокуси еліпсів.
3. Отримати всі можливі трикутники, які можуть бути побудовані на підставі точок БД.
4. По заданому центру і радіусу кола отримати всі точки БД, які лежать усередині кола.
5. На підставі точок БД отримати всі трикутники з рівною площею.
6. По заданих центрах і радіусах двох кіл знайти всі точки, які належать області перетину цих кіл.
7. По заданому радіусу отримати всі точки БД, які можуть бути центрами кіл, що не перетинаються.
8. Отримати всі можливі ромби, які можуть бути побудовані на підставі точок БД.
9. По заданій вершині і параметру p параболи з вертикальною віссю отримати всі точки БД, що належать параболі.
10. По заданих великій і малій осях отримати всі можливі центри еліпсів.
11. По заданих центру, дійсній і уявній осям отримати всі точки БД, що належать гіперболі.
12. Отримати всі можливі квадрати, які можуть бути побудовані на підставі точок БД.

13. По заданому радіусу отримати всі пари центрів кіл, всередину яких потрапляє однакова кількість точок БД.
14. Отримати всі можливі рівносторонні трикутники, які можуть бути побудовані на підставі точок БД.
15. Отримати всі можливі прямокутні трикутники, які можуть бути побудовані на підставі точок БД.
16. По заданій вершині і параметру p параболи з вертикальною віссю отримати всі точки БД, які знаходяться у внутрішній області параболи.
17. Отримати всі можливі паралелограми, які можуть бути побудовані на підставі точок БД.
18. По заданих центрах і радіусах двох кіл знайти всі точки, що належать диз'юнктивній сумі цих кіл.
19. По заданих центру і радіусам двох концентричних кіл знайти всі точки, що належать кільцю, утвореному заданими колами.
20. По заданих центру, великій і малій осям отримати всі точки БД, що знаходяться поза еліпсом.
21. Отримати всі можливі трапеції, які можуть бути побудовані на підставі точок БД.
22. На підставі точок БД отримати всі рівнобедрені трикутники.
23. По заданих центрах і радіусах двох кіл знайти всі точки, що належать об'єднанню цих кіл.
24. Отримати всі можливі трикутники, периметр яких буде менший заданого.
25. По заданій вершині і параметру p параболи з вертикальною віссю отримати всі точки БД, які знаходяться в зовнішній області параболи.

ЗАВДАННЯ № 10

Задача класифікації

Метою є навчитись роботі в Erlang OTP.

Завдання: написати Erlang-програму визначення класу заданого об'єкту за відомими значеннями його ознак. Даних повинно бути достатньо для отримання декількох варіантів рішення.

Приклад виконання індивідуального завдання:

Класифікація стратегій пошуку в просторі станів.

Erlang – програма	**Приклади виконання запитів**
%+ (Evaluation function, Cost function) *%- Strategy* *strategy_classification(0,0)->* *'Depth-first search';* *strategy_classification(0,1)->* *'Breadth-first search';* *strategy_classification(1,0)->* *'Hill-climbing';* *strategy_classification(1,1)->* *'Best-first search'.*	*> guidelines:strategy_classification(1,0).* *'Hill-climbing'* *> guidelines:strategy_classification(1,1).* *'Best-first search'*

Оцінювання стану x при пошуку в просторі станів відбувається за значенням оцінювальної функції

$$f(x) = h(x)+g(x),$$

де h(x) – евристична оцінка шляху від стану x до цільового стану (*Evaluation function)*

g(x) – вартість (довжина) фактичного шляху від початкового стану до стану x *(Cost function)*

В залежності від використання/ не використання h(x) та g(x) визначається стратегія пошуку. В програмі 1 відповідає використанню, 0 – невикористанню відповідного компоненту.

Індивідуальні завдання

1. Класифікація тіл обертань.
2. Класифікація многогранників.
3. Класифікація поверхонь 2-го порядку.
4. Класифікація систем координат космічної геодезії.
5. Класифікація британських кораблів часів Другої світової війни по номеру вимпела.
6. Класифікація ракет за кодовими позначеннями НАТО.
7. Класифікація галактик по послідовності Хабла.
8. Класифікація вантажних автомобілів.
9. Класифікація вин за термінами витримки.
10. Класифікація циліндричних поверхонь.
11. Класифікація многокутників.
12. Класифікація порід коней.
13. Визначення зірок за основною (Гарвардською) спектральною класифікацією.
14. Класифікація кривих 2-го порядку.
15. Класифікація систем координат в елементарній математиці.
16. Класифікація коньяків.
17. Класифікація легкових автомобілів.
18. Класифікація алкогольних напоїв за термінами витримки.
19. Визначення галактик за Гарвардською класифікацією.
20. Класифікація пласких геометричних фігур.
21. Класифікація навколоземних астероїдів.
22. Класифікація комп'ютерних ігор за жанрами.
23. Класифікація кутів в геометрії.
24. Класифікація нафт.
25. Класифікація танків загального призначення за масою.

ЗАВДАННЯ № 11

Розгалужені алгоритми

Метою є навчитись реалізовувати на мові Erlang розгалужені алгоритми.

Завдання: представити на Erlang функцію f(x), задану в індивідуальному завданні.

Приклад виконання індивідуального завдання:

$$f(x) = \begin{cases} 0, x = 0 \\ \sum_{i=1}^{n} x * i, x > 0, \end{cases}$$

Erlang –програма	Приклад виконання запитів
func_value1(0,_N) -> 0; *func_value1(X,N) when X>0 ->* *sum_direct(X,N).* *func_value2(X,N) ->* *if* *X==0 -> 0;* *X>0 -> sum_tail(X,N,0)* *end.* *sum_direct(X,I) when I>0 ->* *(I*X) + sum_direct(X,I-1);* *sum_direct(_X,0) -> 0.* *sum_tail(X,I,Acc) when I>0 ->* *sum_tail(X,I-1, Acc+I*X);* *sum_tail(_X,0,S) -> S.*	*> guidelines:func_value1(1,3).* *6* *> guidelines:func_value2(1,3).* *6*

Реалізації *func_value1/2* та *func_value2/2* еквівалентні. В першому випадку охоронні вирази (Guard) застосовані в голові рівнянь, в другому – в операторі *if*.

В програмі наведено дві версії реалізації підсумовування відповідно з прямою та хвостовою рекурсіями.

Індивідуальні завдання

1. $\begin{cases} x, & \text{якщо } 1<x<2, \\ \prod\limits_{k=1}^{12} kx, & \text{якщо } x>2. \end{cases}$

2. $\begin{cases} 0, & \text{якщо } x<1 \\ x!, & \text{якщо } x>1. \end{cases}$

3. $\begin{cases} k, & \text{якщо } x<0, \\ \sum\limits_{k=1}^{n} (kx+2), & \text{якщо } x>0. \end{cases}$

4. $\begin{cases} x!+x, & \text{якщо } x<55 \\ x, & \text{якщо } x>55. \end{cases}$

5. $\begin{cases} kx^k, & \text{якщо } x<10, \\ \sum\limits_{k=1}^{8} kx, & \text{якщо } x>10. \end{cases}$

6. $\begin{cases} a, & \text{якщо } x<2 \\ (ax)!, & \text{якщо } x>2. \end{cases}$

7. $\begin{cases} (kx)!, & \text{якщо } 0<x<10, \\ x^k, & \text{якщо } x>1. \end{cases}$

8. $\begin{cases} 0, & \text{якщо } x<12 \\ \prod\limits_{k=1}^{10} (kx+a), & \text{якщо } x>12. \end{cases}$

9. $\begin{cases} x^k, & \text{якщо } 7<x<9, \\ kx^k, & \text{якщо } x>9. \end{cases}$

10. $\begin{cases} 5, & \text{якщо } x<100 \\ \sum\limits_{i=1}^{5} (ix-2), & \text{якщо } x>100. \end{cases}$

11. $\begin{cases} c+x, & \text{якщо } x<200, \\ x^n, & \text{якщо } x>200. \end{cases}$

12. $\begin{cases} 0, & \text{якщо } 1<x<2 \\ a+x!, & \text{якщо } x>2. \end{cases}$

13. $\begin{cases} 2, & \text{якщо } x<2, \\ x^n+x^m, & \text{якщо } x>2. \end{cases}$

14. $\begin{cases} 0, & \text{якщо } x<20 \\ a+x!, & \text{якщо } x>20. \end{cases}$

15. $\begin{cases} x^k + 15, & \text{якщо } 1<x<2, \\ \sum\limits_{k=1}^{3} kx^k, & \text{якщо } x>2. \end{cases}$

16. $\begin{cases} x!, & \text{якщо } x>2 \\ \prod\limits_{k=1}^{7} kx, & \text{якщо } 1<x<2. \end{cases}$

17. $\begin{cases} a, & \text{якщо } x>a, \\ \sum\limits_{k=1}^{13} ax^k, & \text{якщо } x<a. \end{cases}$

18. $\begin{cases} k, & \text{якщо } 1<x<k \\ x!, & \text{якщо } x>k. \end{cases}$

19. $\begin{cases} x^n, & \text{якщо } x<n, \\ n, & \text{якщо } x>n. \end{cases}$

20. $\begin{cases} 0, & \text{якщо } x<n \\ \sum\limits_{k=1}^{9} x^k, & \text{якщо } x>n. \end{cases}$

21. $\begin{cases} k, & \text{якщо } x<k, \\ \prod\limits_{k=1}^{15} kx, & \text{якщо } x>k. \end{cases}$

22. $\begin{cases} n + x^n, & \text{якщо } 1<x<2 \\ 100, & \text{якщо } x>2. \end{cases}$

23. $\begin{cases} (x+n)^n, & \text{якщо } x<n, \\ nx^n, & \text{якщо } x>n. \end{cases}$

24. $\begin{cases} k, & \text{якщо } x<k \\ (x+k)!, & \text{якщо } x>k. \end{cases}$

25. $\begin{cases} c + x, & \text{якщо } x>10, \\ x^n, & \text{якщо } x<10. \end{cases}$

ЗАВДАННЯ № 12

Конструювання списків

Метою є навчитись конструювати списки на мові Erlang.

Завдання: отримати список значень заданої в індивідуальному завданні функції. Інтервал значень змінної x, крок в інтервалі і додаткові параметри задаються в запиті.

Приклад виконання індивідуального завдання:

$$f(x) = \sum_{i=1}^{N} i * x$$

Erlang –програма	**Приклад виконання запитів**
%+(X,Final X,Step Size,N) *% -List of Function Values* *func_main(X,Xf,H,N)->* *func_list(X,Xf+H/100,H,N).* *%+(X,Final X,Step Size,N)* *% -List of Function Values* *func_list(X,Xf,H,N)when X=<Xf ->* *[sum_tail(X,N,0)|func_list(X+H,Xf,H,N)];* *func_list(X,Xf,_,_)when X>Xf -> [].*	*6> guidelines:func_main(0,1,0.2,3).* *[0,1.2,2.4,3.6000000000000005,4.8,* *6.0]*

В зовнішній процедурі *func_main/4* вираховується межа інтервалу з урахуванням помилки накопичення. Процедура *func_list/4* безпосередньо будує список значень функції, які розраховуються *sum_ tail /3*, визначеною в програмі до завдання № 11.

Індивідуальні завдання

№ варіанту	Функція y(x)	№ варіанту	Функція y(x)
1	$\sum_{i=1}^{N} \frac{\cos(x+2i)}{i^2+1} - 2x^2$	14.	$x + \frac{1}{x^N} + \prod_{i=1}^{N} \cos(* x$

2	$(\sum_{i=1}^{N}\frac{1}{i^2})x \Big/ \sum_{k=2}^{M}\frac{1}{k^2-1}$	15.	$\frac{tg^2 x}{\sum_{i=1}^{N} i * x}$
3	$(\sum_{i=1}^{N}\frac{1}{x^2+i})+\sin x^2$	16.	$\frac{1+x}{1+x^3} * \prod_{i=3}^{N}\left(i^{\frac{1}{2}}-x\right)$
4	$\sum_{i=2}^{N}\frac{x^4-x^2}{i^2-1}\Big/x$	17.	$\sqrt{x+2} * \sum_{i=1}^{N}\frac{x^i}{i}$
5	$(\prod_{i=1}^{N}(i^2+x))x^2$	18.	$\frac{\prod_{i=1}^{N}(x^i+N)}{x+N}$
6	$\frac{1}{1+\cos x}\prod_{i=1}^{N}\frac{i^{-2}+0{,}5}{3}$	19.	$x^N * \sum_{i=1}^{N}\frac{i}{x+i}$
7.	$\left(\sum_{i=1}^{N}\frac{x^2}{i+x}\right)+tgx$	20	$\frac{x}{N}+\sum_{i=1}^{N}\frac{x+iM}{2i+1}$
8.	$\cos x^2 * \prod_{i=1}^{N}(x^3+2.3*i)$	21	$\sqrt{x+1}\Big/\sum_{i=1}^{N} i\sin x$
9.	$\sqrt{x^2+5*x+2} * \sum_{i=1}^{N}\frac{1}{i*x^2}$	22	$\frac{N}{x}+\prod_{i=1}^{N}\frac{N+i+x}{i}$
10.	$\frac{x}{(x+x)^2}\prod_{i=3}^{N}(x+x)$	23	$\cos Nx\sum_{i=1}^{N}\frac{i}{N\sin x}$
11.	$(\sum_{i=1}^{N}\frac{1}{x+i})+tg(x^2)$	24	$N!\prod_{i=1}^{N}(x+i)$
12.	$\frac{\sin(x)}{x} * \prod_{i=2}^{N}\frac{(x^i-x)}{i}$	25	$\sqrt{x}+\sum_{i=1}^{N}x^i$
13.	$\sum_{i=1}^{N}\frac{x+i}{i!}-\cos x$		

ЗАВДАННЯ № 13

Обробка елементів списків

Метою є навчитись обробляти елементи списків на мові Erlang.

Завдання: написати програму рішення вправи, вказаної в індивідуальному завданні.

Приклад виконання індивідуального завдання:

замінити n-й елемент списку на останній. Число n задається в запиті.

Erlang –програма	**Приклад виконання запитів**
% ver.1 *%+(List,N)* *%-New List* *replace1(Xs,N)->* *[_|L] = part2(Xs,N),* *part1(Xs,N)++[lists:last(Xs)|L].* *%+(List,N)* *%-Part before nth item* *part1([X|Xs],I)when I>1 ->* *[X|part1(Xs,I-1)];* *part1(_,1)->[].* *%+(List,N)* *%-Part after nth item* *part2([_X|Xs],I)when I>1 ->* *part2(Xs,I-1);* *part2(L,1)->L.* *% ver.2* *%+(List,N)* *%-New List* *replace2([X|Xs],I) when I>1 ->* *[X|replace2(Xs,I-1)];* *replace2([_Xn|Xs],1)->	*>guidelines:replace1([1,2,3,4,5],3).* *[1,2,5,4,5]* *>guidelines:replace2([1,2,3,4,5],3).* *[1,2,5,4,5]*

{Xz,Tail} = replace21(Xs,[]), *[Xz|Tail].* *%+(List,Accumulator* *%-{Last Item, List after Xn}* *replace21([Xz],Acc)->* *{Xz,lists:reverse([Xz|Acc])};* *replace21([X|Xs],Acc)->* *replace21(Xs,[X|Acc]).*	

Процедура *replace1/2* збирає в необхідній послідовності частину списку до n-го елемента, останній елемент та частину списку після n-го елемента. Перша версія більш зрозуміла, але потребує на n/2 більше ітерацій ніж друга.

Процедура *replace2/2* відтворює випадок перезапису в новий список всіх елементів до n-го. Процедура *replace21/2* повертає кортеж з останнього елемента та решти заданого списку після n-го елементу.

Індивідуальні завдання

1. Видалити із списку елементи, що повторюються, залишивши один екземпляр.
2. Видалити передостанній і останній елементи списку і вставити їх відповідно першим і другим.
3. Додати в кінець заданого числового списку елемент, який є сумою 1-го і останнього елементів.
4. Видалити із списку всі елементи, що стоять на парних позиціях.
5. Заданий числовий список. Видалити з нього всі непарні числа.
6. Замінити n-й елемент списку на перший. Число n задається в запиті.
7. Поміняти місцями перший і останній елементи списку.
8. Знайти «серединний» елемент списку непарної довжини.
9. Заданий список непарної довжини. Поміняти місцями елемент, що стоїть до «серединного» елементу, з елементом, що стоїть після «серединного» елементу.

10. Видалити перші і останні три елементи заданого списку.
11. Заданий числовий список. Отримати список, в якому залишаться тільки цілі числа.
12. Вставити окремо заданий термін після «серединного» елементу списку непарної довжини.
13. Видалити із списку всі елементи, ідентичні заданому зразку, не залишаючи жодного екземпляра.
14. Видалити із списку всі елементи, що стоять на позиціях, кратних трьом.
15. Заданий числовий список. Отримати список, в якому залишаться лише парні числа.
16. Заданий числовий список непарної довжини. Знайти суму трьох його «серединних» елементів.
17. Додати в кінець заданого числового списку елемент, який є сумою 1-го і «серединного» елементів.
18. Додати в кінець заданого списку його другий і п'ятий елементи і видалити передостанній елемент.
19. Поміняти місцями n-й і (n+1)-й елементи списку. Число n задається в запиті.
20. Видалити із списку елементи, що все повторюються, не залишаючи жодного екземпляра.
21. Замінити кожен третій елемент заданого списку на останній.
22. Скласти список, що складається з елементів заданого списку, які більше обох своїх сусідів.
23. Отримати список, що складається з елементів заданого списку, що не повторюються.
24. Отримати список, що складаються з парних елементів заданого числового списку.
25. Заданий числовий список. Отримати список, в якому залишаться лише від’ємні числа.

ЗАВДАННЯ № 14

Виокремлення частин списків

Метою є навчитись обробляти списки на мові Erlang.

Завдання: написати програму рішення вправи, вказаної в індивідуальному завданні.

Приклад виконання індивідуального завдання:

задано список, довжина якого кратна 4. Видалити третю чверть списку, а елементи другої розташувати в порядку, зворотному заданому.

Erlang -програма	**Приклад виконання запитів**
% ver 1 *%+List* *%-New List* *% ver.1* *list_pro_main1(L)->* *K = length(L) div 4,* *K2 = K+1,* *K3 = 2*K+1,* *part1(part1(L,K3),K2)++lists:reverse(part2(part1(L,K3),K2))++part2(part2(L,K3),K2).* *% ver.2* *list_pro_main2(L)->* *K = length(L) div 4,* *K2 = K+1,* *K3 = 2*K+1,* *K4 = 3*K+1,* *list_pro(L,1,K2,K3,K4,[]).* *list_pro([X|Xs],I,K2,K3,K4,Acc) when I<K2 ->* *% 1st quarter* *[X|list_pro(Xs,I+1,K2,K3,K4,Acc)];* *list_pro([X|Xs],I,K2,K3,K4,Acc)* *when I>=K2,I<K3 ->* *% 2nd quarter*	*>guidelines:list_pro_main1([1,2,3,a,b,c,x,x,x,11,22,33]).* *[1,2,3,c,b,a,11,22,33]*

list_pro(Xs,I+1,K2,K3,K4,[X|Acc]); *list_pro([_X|Xs],I,K2,K3,K4,[X|Acc])* *when I>=K3, I<K4 ->* *% 3rd quarter* *[X|list_pro(Xs,I+1,K2,K3,K4,Acc)];* *list_pro(Xs,K4,_K2,_K3,K4,_Acc) -> Xs.* *% 4th quarter*	

Наведено 2 версії розв'язання задачі. В першій – список ділиться на відповідні частини *part1/2* та *part2/2*, визначені в прикладі до завдання № 13. Друга чверть обертається *reverse/2*. Злиття нового списку в потрібному порядку здійснюється оператором ++.

В другій версії представлено оптимальне рішення задачі в одній процедурі *lists_pro/6* за 3/4n (де n - довжина списку) ітерацій. Під час проходження по списку:

- в першій чверті - елементи відразу переписуються в новий список,
- в другій чверті - накопичуються в акумуляторі в зворотному порядку,
- в третій чверті – з акумулятора переписуються в новий список,
- на першому елементі четвертої чверті рекурсія зупиняється і решта заданого списку повністю переноситься в новий.

Індивідуальні завдання

1. Задано список, довжина якого кратна 4. Видалити 1-у чверть списку, а елементи 3-ої розташувати в порядку, зворотному заданому.
2. Задано список, довжина якого кратна трьом. Отримати список, в якому перша і друга третини поміняються місцями.
3. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена остання третина.
4. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена перша третина, а елементи останньої третини розташовані в порядку, зворотному заданому.

5. Задано список, довжина якого кратна 4. Видалити останню чверть списку, а елементи першої розташувати в порядку, зворотному заданому.
6. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з третьої третини першого списку, другої третини другого і першої третини третього.
7. Задано 2 списки, довжина яких кратна 2. Отримати список, складений з половинок заданих списків в наступній послідовності: 1-а половина першого списку, 1-а половина другого списку, 2-а половина першого списку, 2-а половина другого списку.
8. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з першої третини першого списку, другої третини другого і третьої третини третього.
9. Задано список, довжина якого кратна трьом. Отримати список, в якому перша і остання третини збережуться, а елементи другої третини знаходитимуться в порядку, зворотному заданому.
10. Задано список, довжина якого кратна трьом. Отримати список, в якому елементи другої третини знаходяться в порядку, зворотному заданому.
11. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена остання третина.
12. Задано список, довжина якого кратна 4. Видалити 3-у чверть списку, а елементи 2-ої розташувати в порядку, зворотному заданому.
13. Задано список парної довжини. Отримати список, в якому перша і друга половини поміняються місцями.
14. Задано список, довжина якого кратна трьом. Отримати список, в якому видалена друга третина.
15. Задано список парної довжини. Отримати список, в якому перша і друга половини поміняються місцями, причому елементи другої половини заданого списку йтимуть в порядку, зворотному заданому.
16. Задано 2 списки, довжина яких кратна 2. Отримати список, складений з половинок заданих списків в наступній послідовності: 2-а половина першого

списку, 2-а половина другого списку, 1-а половина першого списку, 1-а половина другого списку.

17. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з першої третини першого списку, третьої третини другого і другої третини третього. Елементи останньої третини сформованого списку повинні розташовуватися в порядку, зворотному заданому.
18. Задано список, довжина якого кратна 4. Видалити 4-у чверть списку, а елементи 1-ої розташувати в порядку, зворотному заданому.
19. Задано список, довжина якого кратна трьом. Отримати список, в якому остання третина зберігається, а перша і друга міняються місцями. Причому елементи другої третини знаходитимуться в порядку, зворотному заданому.
20. Задано список, довжина якого кратна трьом. Отримати список, в якому елементи останньої третини знаходяться в порядку, зворотному заданому.
21. Задано 2 списки парної довжини. Отримати список, складений з другої половини першого заданого списку і першої половини другого.
22. Задано список, довжина якого кратна 4. Видалити 2-у чверть списку, а елементи 4-ої розташувати в порядку, зворотному заданому.
23. Задано список, довжина якого кратна трьом. Отримати список, в якому друга і остання третини поміняються місцями.
24. Задано 3 списки, довжина яких кратна трьом. Отримати список, складений з другої третини першого списку, першої третини другого і третьої третини третього.
25. Задано 2 списки, довжина яких кратна 2. Отримати список, складений з половинок заданих списків в наступній послідовності: 1-а половина першого списку, 1-а половина другого списку, 2-а половина першого списку, 2-а половина другого списку. Причому елементи других половин повинні бути розташовані в зворотному порядку по відношенню до заданого.

ЗАВДАННЯ № 15

Обробка списку списків

Метою є навчитись обробляти списки списків на мові Erlang.

Завдання: виконати перетворення матриці або розрахунок її заданої характеристики. Якщо в завданні мова йде про діагоналі, задану матрицю вважати квадратною.

Приклад виконання індивідуального завдання:

Задана квадратна матриця невідомого розміру. Визначити суму елементів під побічною діагоналлю матриці (включно з елементами діагоналі).

Erlang –програма	**Приклади виконання запитів**
matrix_data(1)-> *[[1,1,1,1,1],* *[1,1,1,1,1],* *[1,1,1,1,1],* *[1,1,1,1,1],* *[1,1,1,1,1]];* *matrix_data(2)->* *[[1,1,1,1,0],* *[1,1,1,0,0],* *[1,1,0,0,0],* *[1,0,0,0,0],* *[0,0,0,0,0]].* *%+ Test#* *matrix_main(K)->* *Ms = matrix_data(K),* *matrix_pro(Ms,length(Ms),0).* *%+Matrix, Rows Counter, Accumulator,* *%-Sum* *matrix_pro([R|Rs],I,Acc)->* *matrix_pro(Rs,I-1,* *Acc+row_pro(R,1,I));*	*> guidelines:matrix_main(1).* *15* *> guidelines:matrix_main(2).* *0*

matrix_pro([],_,S)->S. *%+Row, Counter, N* *%-Row Sum* *row_pro([_|Xs],I,N) when I<N ->* *row_pro(Xs,I+1,N);* *row_pro([X|Xs],I,N) when I>=N ->* *X+ row_pro(Xs,I+1,N);* *row_pro([],_,_)->0.*	

В процедурі *matrix_pro/3* рекурсія відбувається по рядках матриці. На кожній ітерації обчислюється сума відповідних елементів поточного рядку – процедура *row_pro/3*. В *matrix_pro/3* реалізовано пряму рекурсію, а в *row_pro/3* – хвостову.

Індивідуальні завдання

1. Задана матриця невідомих розмірів. Отримати матрицю, в якій крайні елементи головної діагоналі поміняються місцями.
2. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій головна і побічна діагоналі поміняються місцями.
3. Задана квадратна матриця невідомого розміру. Отримати суму елементів, розташованих вище за головну діагональ матриці.
4. Задана квадратна матриця невідомого розміру. Отримати суму елементів третього квадранта матриці.
5. Задана матриця невідомих розмірів. Отримати матрицю, в якої половини головної діагоналі поміняються місцями.
6. Задана матриця розмірів MXN. Отримати матрицю, в якій поміняються місцями (M,1) -й і (1,1) -й елементи.
7. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на половинах діагоналей, розташованих праворуч від центру.
8. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на побічній діагоналі.

9. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій крайні елементи побічної діагоналі поміняються місцями.
10. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій поміняються місцями верхні половини головної і побічної діагоналей.
11. Задана матриця невідомих розмірів. Отримати суму центрального і всіх елементів, що оточують його.
12. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на головній діагоналі.
13. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями половини головної і побічної діагоналей, що знаходяться зліва від центру.
14. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій крайні елементи головної діагоналі поміняються місцями з сусідніми по діагоналі елементами.
15. Задана матриця невідомих розмірів. Отримати суму елементів першого квадранта матриці.
16. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій крайні елементи побічної діагоналі поміняються місцями з сусідніми по діагоналі елементами.
17. Задана матриця розмірів MXN. Отримати матрицю, в якій поміняються місцями (M,n) -й і (1,n) -й елементи.
18. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями центральний стовпчик і центральний рядок матриці.
19. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями половини головної і побічної діагоналей, що знаходяться праворуч від центру.
20. Задана матриця невідомих розмірів. Отримати суму елементів другого квадранта матриці.
21. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на половинах діагоналей, розташованих вище за центр.

22. Задана квадратна матриця невідомого розміру. Отримати матрицю, в якій міняються місцями нижні половини головної і побічної діагоналей.
23. Задана квадратна матриця невідомого розміру. Отримати суму елементів, що знаходяться на повічній діагоналі.
24. Задана квадратна матриця невідомого розміру. Отримати суму елементів, розташованих вище за побічну діагональ матриці.
25. Задана матриця невідомих розмірів. Отримати суму елементів, що знаходяться на обох діагоналях.

ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ № 16

Використання генераторів списків

Метою є навчитись використовувати генератори списків для обробки формалізованих даних.

Завдання: данні, необхідні для виконання індивідуального завдання, представити у вигляді списку відповідних кортежів. Розв'язати завдання з використанням генераторів списків.

Приклад виконання індивідуального завдання:

два боксери можуть зустрітися на ринзі, якщо вони в одній ваговій категорії і у них однаковий спортивний розряд.

Erlang –програма	**Приклади виконання запитів**
% {Name, Wight Category, Class} *db_boxers() ->* *[{adam, heavyweight, 1},* *{bob, light_weight, 1},* *{dick, middle_weight, 1},* *{john, heavyweight, 1},* *{mike, light_weight, 2},* *{nick, light_weight, 1}].* *on_ring() ->* *DB = db_boxers(),* *L = [{Wight, Class,* *[Name||{Name,W,C} <- DB,* *Wight == W, Class == C]} ||* *{_,Wight,Class} <- DB],* *del_dbles_mod(L).* *del_dbles_mod([{W,C,L}|T]) ->* *Tclear = del_examples_mod(T,{W,C}),* *[{W,C,L}|del_dbles_mod(Tclear)];* *del_dbles_mod([]) -> [].*	*> guidelines:on_ring().* *[{heavyweight,1,[adam,john]},* *{light_weight,1,[bob,nick]},* *{middle_weight,1,[dick]},* *{light_weight,2,[mike]}]*

del_examples_mod([],_) -> []; *del_examples_mod([{W,C,L}|Xs],Obj) ->* *case {W,C} == Obj of* *true -> del_examples_mod(Xs,Obj);* *false -> {W,C,L}|* *del_examples_mod(Xs,Obj)]* *end.*	

Розв'язання задачі спершу потребує визначення структур даних. Для представлення інформації про спортсмена в БД використовується кортеж *{Ім'я, Вагова категорія,Спортивний розряд}*. Запис рішень поставленого завдання відбувається за форматом наступного кортежу: *{Вагова категорія,Спортивний розряд, Список імен боксерів, які відповідають першим двом характеристикам}*.

В функції *on_ring/0* генератор створює список відповідей. В функції *del_dbles_mod/1* видаляються дублікати відповідей. На кожній ітерації з хвоста списку функцією *del_examples_mod/2* видаляються дублікати списків, які відповідають поточній комбінації значень *{Вагова категорія,Спортивний розряд}*.

Індивідуальні завдання

1. Два спортсмени можуть брати участь в одному і тому ж змаганні, якщо вони займаються одним і тим же видом спорту і мають один і той же розряд.
2. У добірку картин для виставки можуть увійти три твори, що відносяться до одного художнього напряму і до однієї епохи створення.
3. У акваріум можна одночасно помістити риб трьох різновидів, якщо всі вони належать або не належать до хижаків.
4. Два солдати термінової служби можуть демобілізуватися разом, якщо вони - земляки одного призиву.
5. Дві люди є братами, якщо вони - чоловіки і мають загальних батьків.
6. Дві кості доміно можуть бути встановлені поряд, якщо співпадає кількість крапок на дотичних половинках.

7. Два прикладні програмні продукти сумісні, якщо вони відкомпілювалися одним компілятором.
8. У збірку можуть увійти 3 літературних твори одного напряму, написані на одній мові.
9. Два лікарські препарати взаємозамінні, якщо вони мають одну й ту саму діючу речовину та однакові протипоказання до застосування.
10. Два спортсмени можуть увійти до команди, якщо вони мають однаковий спортивний розряд і займаються різними видами спорту.
11. Дві футбольні команди різних країн можуть зустрітися в Єврокубку, якщо вони зайняли одне з перших чотирьох місць в національному чемпіонаті або виграли Кубок своєї країни.
12. Три квітки можуть скласти букет, якщо їх цвітіння відбувається в один і той же місяць.
13. Двоє тварин можуть одночасно знаходитися поряд на водопої, якщо обидві належать до травоїдних або м'ясоїдних.
14. У концерті можуть брати участь три музичні групи одного стилю і одного рівня виконання.
15. До збірки завдань контрольної роботи можуть увійти три завдання, які відносяться до різних розділів однієї і тієї ж навчальної дисципліни.
16. На ділянку можна посадити дерева трьох найменувань, якщо всі вони пристосовані до одного і тому ж клімату і є або листовими, або хвойними, або плодовими.
17. У стопку можна покласти 3 гральних карти різної гідності, але однакової масті.
18. Два різних компонента одягу можна одягнути одночасно, якщо вони відносяться до одного сезону і відповідають одному і тому ж типу погоди.
19. Двоє дітей одного віку можуть ходити в одну групу дитячого садку, якщо вони живуть в одному будинку.
20. У чвертьфіналі Ліги чемпіонів можуть зустрітися дві футбольних команди з різних підгруп, якщо вони зайняли перше або друге місце в своїй підгрупі.

21. Двоє тварин можуть скласти пари, якщо вони протилежної статі і однієї породи. (Створити БД по котах або собаках.)
22. Три квітки можуть скласти букет, якщо вони мають однакові колір та сорт.
23. Фруктовий коктейль повинен складатися з трьох фруктів різних кольорів, які дозрівають в один і той же місяць.
24. Дві людини є близнятами, якщо у них одні і ті ж батьки і вони народилися в один і той же день.
25. У стопку можна покласти 3 гральних карти різної масті, але однакової гідності.

ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ № 17

Фільтрація списку

Метою є навчитись вирішувати базові задачі обробки списків усіма можливими способами.

Завдання: написати три версії вирішення вправи, вказаної в індивідуальному завданні. Відповідно до версій необхідно:

1. умову фільтрації представити охоронним виразом для вибору відповідного рівняння на основі співставленні зі зразком;
2. використати case-вираз;
3. використати BIF lists:filter/2;
4. використати генератор списку.

Приклад виконання індивідуального завдання:

Задано числовий список. Отримати список, в якому залишаться лише цілі числа.

Erlang –програма	**Приклади виконання запитів**
% ver.1 *only_integers1([X|Xs]) when is_integer(X)->* *[X|only_integers1(Xs)];* *only_integers1([X|Xs]) when not(is_integer(X))->* *only_integers1(Xs);* *only_integers1([])->[].* *% ver.2* *only_integers2([X|Xs])->* *case is_integer(X) of* *true ->* *[X|only_integers2(Xs)];* *_ -> only_integers2(Xs)*	*>guidelines:only_integers1([1.0,2,3,0.0,100]).* *[2,3,100]*

end; *only_integers2([])->[].* *% ver.3* *only_integers3(Xs)->* *lists:filter(fun (X) -> is_integer(X) end, Xs).* *% ver.4* *only_integers4(Xs)->* *[X||X <- Xs,is_integer(X)].*	

Нумерація функцій *only_integersN/1* відповідає нумерації способів вирішення завдання. Оскільки всі функції повертають одне й те саме рішення, наведено тільки один приклад запиту.

Індивідуальні завдання

1. Видалити букви z, w із заданого списку латинських букв.
2. Задано числовий список. Отримати два списки, в першому з яких будуть тільки парні числа, а в другому – тільки непарні.
3. Задано числовий список. Змінити знак всіх від'ємних чисел, які знаходяться в діапазоні [-1,0].
4. Задано числовий список. Видалити з нього всі від'ємні числа.
5. Задано числовий список. Всі числа, які більші 100, зменшити на 100.
6. Задано список букв латинського алфавіту. Видалити з нього всі букви, які за алфавітом знаходяться між h та n.
7. Задано числовий список. Отримати два списки, в першому з яких будуть тільки додатні числа, а в другому – тільки від'ємні.
8. Задано числовий список. Змінити знак всіх додатних чисел списку, які менші 10.
9. Задано числовий список. Отримати список, в якому залишаться тільки парні числа.
10. Задано список результатів вимірювань кутів. Видалити ті значення, sin яких дорівнює 0.

11. Задано числовий список. Отримати два списки, в першому з которих будуть тільки цілі числа, а в другому – тільки дійсні.
12. Задано числовий список. Видалити з нього всі числа 0, 10, 100.
13. Задано числовий список. Отримати список позицій, на яких розташовані від'ємні числа.
14. Видалити зі списка всі елементи, які ідентичні заданому зразку.
15. Задано список букв. Отримати два списки, в першому з которих будуть тільки голосні, а в другому – тільки приголосні.
16. Задано числовий список. Видалити з нього всі числа в діапазоні від 0 до 10.
17. Задано числовий список. Подвоїти всі додатні числа та видалити від'ємні.
18. Знайти кількість входжень заданого зразка в заданий список.
19. Задано числовий список. Видалити з нього всі непарні числа.
20. Задано числовий список. Отримати список, в якому залишаться тільки цілі числа.
21. Задано числовий список. Отримати два списки, в першому з которих будуть тільки додатні числа, а в другому – тільки від'ємні.
22. Задано числовий список. Видалити з нього всі букви, які за алфавітом знаходяться після заданої букви.
23. Задано список букв. Видалити з нього всі приголосні.
24. Задано числовий список. У всіх від'ємних числах замінити знак на протилежний.
25. Задано числовий список. Отримати список, в якому залишаться тільки числа, кратні 3.

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Братко И. Агоритмы искусственного интеллекта на языке PROLOG: Пер с англ. – М.: Издательский дом «Вильямс», 2004. – 640с.
2. Стерлинг Л., Шапиро Э. Искусство программирования на языке Пролог: Пер. с англ. – М.: Мир, 1990. – 235с.
3. Томпсон С., Чезарини Ф. Программирование в Erlang. – М.: ДМК Пресс, 2012. – 488 с.
4. Wielemaker, J., SWI-Prolog 5.4 // Reference Manual. – http://www.swi-prolog.org
5. Душкин Р.В. Основы функционального программирования // Учебник по функциональному программированию. – Режим доступа: http://ru.wikibooks.org/wiki/Основы_функционального_ программирования
6. Малпас Дж. Реляционный язык Пролог и его применение: Пер. с англ. – М.: Наука, 1990. – 464с.
7. Шрайнер П.А. Основы программирования на языке Пролог // Интернет Университет информационных технологий. – Режим доступа: http://www.intuit.ru/department/pl/plprolog/
8. Rowe, N., Artificial Intelligence through Prolog. – Mass.: Addison–Wesley, 1986. – 451p.
9. Дехтяренко И.А. Декларативное программирование // Учебное пособие – Режим доступа: http://www.softcraft.ru/paradigm/dp/index.shtml
10. Люгер Дж. Искусственный интеллект: стратегии и методы решения сложных проблем. – М.:Издательский дом "Вильямс", 2004. – 864с.
11. Armstrong, J., Programming Erlang. Software for concurrent world. – Raleigh, North Carolina; Dallas,Texas: The Pragmatic Bookshelf, 2007. – 515p.

ІНФОРМАЦІЙНІ РЕСУРСИ

1. SWI-Prolog: comprehensive free Prolog environment / Download, Documentation, Tutorials, Community. URL: http://www.swi-prolog.org/
2. Erlang Programming Language / Download Erlang/OTP, News, Articles, Documentation, etc. URL: http://www.erlang.org/